Homework 1

Jialu Wang

September 12, 2022

Problem 1:

$$\boldsymbol{c}_{Katz} = \beta(\boldsymbol{I} - \alpha\boldsymbol{A})^{-1}\overrightarrow{\boldsymbol{1}} \tag{1}$$

The matrix $(\boldsymbol{I} - \alpha\boldsymbol{A})^{-1}$ diverges if det $(\boldsymbol{I} - \alpha\boldsymbol{A})^{-1}$ passes zero, i.e.:

$$\det(\boldsymbol{A} - \alpha^{-1}\boldsymbol{I}) = 0 \tag{2}$$

when $\alpha^{-1} = \lambda_1$, the determinant passes zero. λ_1 is the largest eigenvalue of $\boldsymbol{A}$. To ensure the convergence, $\alpha < \lambda_1^{-1}$

Problem 2:

The number of walks of length 1 between ν_i and ν_j is denoted by walk A_{ij}:

$$N_{ij}^{(1)} = A_{ij} \tag{3}$$

A common neighbour is a node that connects both ν_i and ν_j, i.e. it forms a walk of size 2, starting and ending at ν_i and ν_j, connecting by this neighbour. Since it's not a directed graph:

$$N_{ij}^{(2)} = \sum_{k=1}^{n} A_{ik}A_{kj} = [A^2]_{ij} \tag{4}$$

$[A^2]_{ij}$ is the number of common neighbours.

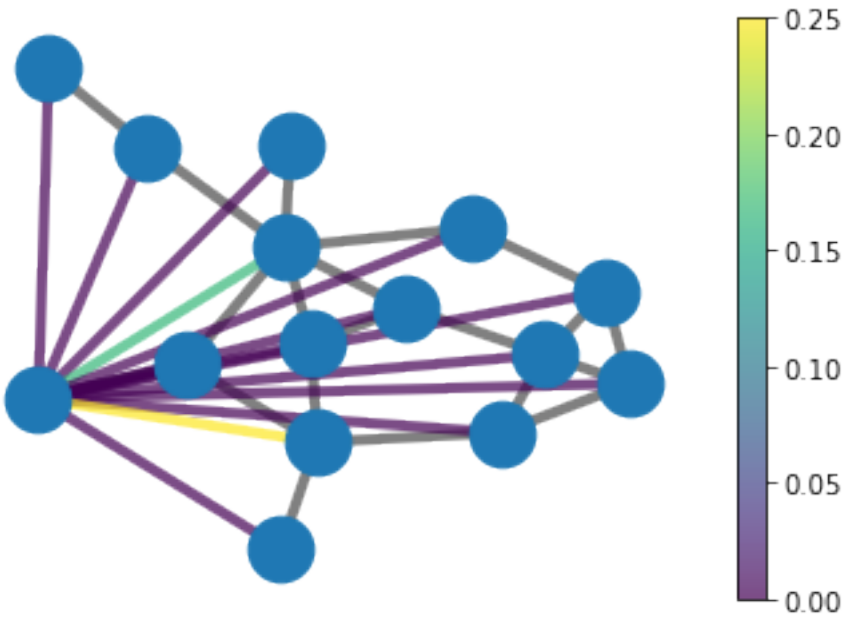

Figure 1: Similarity between 'Ginori' and other families.

Problem 3:

Please see the notebook HW1.ipynb for code. I also attach a pdf after this page.

In [1]:
```
import networkx as nx
import matplotlib.pyplot as plt
import numpy as np
from networkx.algorithms import bipartite
from networkx.generators.random_graphs import erdos_renyi_graph
import copy
```

In [2]:
```
# -- Initialize graphs
seed = 30
G = nx.florentine_families_graph()
nodes = G.nodes()

layout = nx.spring_layout(G, seed=seed)
```

In [3]:
```
Nnodes = len(nodes)

# map node index to its name
MapNode = {}
for i in range(Nnodes):
    MapNode[i] = list(nodes)[i]

# adjacency matrix
A = np.zeros((Nnodes, Nnodes))

for i in range(Nnodes):
    for j in range(i+1, Nnodes):
        if ((MapNode[i], MapNode[j]) in list(G.edges)) or ((MapNode[j], M
            A[i,j], A[j,i] = 1, 1
```

In [4]:
```
# compute commen neighbour and total distinct
Common = np.zeros((Nnodes, Nnodes))
Total = np.zeros((Nnodes, Nnodes))

# common neighbours = A[i]@A[j]
for i in range(Nnodes):
    for j in range(i+1, Nnodes):
        Common[i,j] = Common[j,i] = A[i]@A[j]


# total neighbours = i neighbours + j neibours - common neighbours
for i in range(Nnodes):
    for j in range(i+1, Nnodes):
        Total[i,j] = Total[j,i] = sum(A[i]) + sum(A[j]) - Common[i,j]
```

In [5]:
```
# compute similarity
Similar = np.zeros((Nnodes, Nnodes))

# add tuples containing similarity
# similarity = common/total
pred = []
for i in range(Nnodes):
    for j in range(i+1, Nnodes):
```

```
        Similar[i,j] = Similar[j,i] = Common[i,j]/Total[i,j]
        pred.append((MapNode[i], MapNode[j], Similar[i,j]))
```

In [6]:

```
# -- keep a copy of edges in the graph
old_edges = copy.deepcopy(G.edges())
```

In [7]:

```
# -- add new edges representing similarities.
new_edges, metric = [], []
for u, v, p in pred:
    G.add_edge(u, v)
    print(f"({u}, {v}) -> {p:.8f}")
    new_edges.append((u, v))
    metric.append(p)
```

```
(Acciaiuoli, Medici) -> 0.00000000
(Acciaiuoli, Castellani) -> 0.00000000
(Acciaiuoli, Peruzzi) -> 0.00000000
(Acciaiuoli, Strozzi) -> 0.00000000
(Acciaiuoli, Barbadori) -> 0.50000000
(Acciaiuoli, Ridolfi) -> 0.33333333
(Acciaiuoli, Tornabuoni) -> 0.33333333
(Acciaiuoli, Albizzi) -> 0.33333333
(Acciaiuoli, Salviati) -> 0.50000000
(Acciaiuoli, Pazzi) -> 0.00000000
(Acciaiuoli, Bischeri) -> 0.00000000
(Acciaiuoli, Guadagni) -> 0.00000000
(Acciaiuoli, Ginori) -> 0.00000000
(Acciaiuoli, Lamberteschi) -> 0.00000000
(Medici, Castellani) -> 0.12500000
(Medici, Peruzzi) -> 0.00000000
(Medici, Strozzi) -> 0.11111111
(Medici, Barbadori) -> 0.00000000
(Medici, Ridolfi) -> 0.12500000
(Medici, Tornabuoni) -> 0.12500000
(Medici, Albizzi) -> 0.00000000
(Medici, Salviati) -> 0.00000000
(Medici, Pazzi) -> 0.16666667
(Medici, Bischeri) -> 0.00000000
(Medici, Guadagni) -> 0.25000000
(Medici, Ginori) -> 0.16666667
(Medici, Lamberteschi) -> 0.00000000
(Castellani, Peruzzi) -> 0.20000000
(Castellani, Strozzi) -> 0.16666667
(Castellani, Barbadori) -> 0.00000000
(Castellani, Ridolfi) -> 0.20000000
(Castellani, Tornabuoni) -> 0.00000000
(Castellani, Albizzi) -> 0.00000000
(Castellani, Salviati) -> 0.00000000
(Castellani, Pazzi) -> 0.00000000
(Castellani, Bischeri) -> 0.50000000
(Castellani, Guadagni) -> 0.00000000
(Castellani, Ginori) -> 0.00000000
(Castellani, Lamberteschi) -> 0.00000000
(Peruzzi, Strozzi) -> 0.40000000
(Peruzzi, Barbadori) -> 0.25000000
(Peruzzi, Ridolfi) -> 0.20000000
(Peruzzi, Tornabuoni) -> 0.00000000
```

```
(Peruzzi, Albizzi) -> 0.00000000
(Peruzzi, Salviati) -> 0.00000000
(Peruzzi, Pazzi) -> 0.00000000
(Peruzzi, Bischeri) -> 0.20000000
(Peruzzi, Guadagni) -> 0.16666667
(Peruzzi, Ginori) -> 0.00000000
(Peruzzi, Lamberteschi) -> 0.00000000
(Strozzi, Barbadori) -> 0.20000000
(Strozzi, Ridolfi) -> 0.00000000
(Strozzi, Tornabuoni) -> 0.16666667
(Strozzi, Albizzi) -> 0.00000000
(Strozzi, Salviati) -> 0.00000000
(Strozzi, Pazzi) -> 0.00000000
(Strozzi, Bischeri) -> 0.16666667
(Strozzi, Guadagni) -> 0.14285714
(Strozzi, Ginori) -> 0.00000000
(Strozzi, Lamberteschi) -> 0.00000000
(Barbadori, Ridolfi) -> 0.25000000
(Barbadori, Tornabuoni) -> 0.25000000
(Barbadori, Albizzi) -> 0.25000000
(Barbadori, Salviati) -> 0.33333333
(Barbadori, Pazzi) -> 0.00000000
(Barbadori, Bischeri) -> 0.00000000
(Barbadori, Guadagni) -> 0.00000000
(Barbadori, Ginori) -> 0.00000000
(Barbadori, Lamberteschi) -> 0.00000000
(Ridolfi, Tornabuoni) -> 0.20000000
(Ridolfi, Albizzi) -> 0.20000000
(Ridolfi, Salviati) -> 0.25000000
(Ridolfi, Pazzi) -> 0.00000000
(Ridolfi, Bischeri) -> 0.20000000
(Ridolfi, Guadagni) -> 0.16666667
(Ridolfi, Ginori) -> 0.00000000
(Ridolfi, Lamberteschi) -> 0.00000000
(Tornabuoni, Albizzi) -> 0.50000000
(Tornabuoni, Salviati) -> 0.25000000
(Tornabuoni, Pazzi) -> 0.00000000
(Tornabuoni, Bischeri) -> 0.20000000
(Tornabuoni, Guadagni) -> 0.00000000
(Tornabuoni, Ginori) -> 0.00000000
(Tornabuoni, Lamberteschi) -> 0.33333333
(Albizzi, Salviati) -> 0.25000000
(Albizzi, Pazzi) -> 0.00000000
(Albizzi, Bischeri) -> 0.20000000
(Albizzi, Guadagni) -> 0.00000000
(Albizzi, Ginori) -> 0.00000000
(Albizzi, Lamberteschi) -> 0.33333333
(Salviati, Pazzi) -> 0.00000000
(Salviati, Bischeri) -> 0.00000000
(Salviati, Guadagni) -> 0.00000000
(Salviati, Ginori) -> 0.00000000
(Salviati, Lamberteschi) -> 0.00000000
(Pazzi, Bischeri) -> 0.00000000
(Pazzi, Guadagni) -> 0.00000000
(Pazzi, Ginori) -> 0.00000000
(Pazzi, Lamberteschi) -> 0.00000000
(Bischeri, Guadagni) -> 0.00000000
(Bischeri, Ginori) -> 0.00000000
(Bischeri, Lamberteschi) -> 0.33333333
(Guadagni, Ginori) -> 0.25000000
(Guadagni, Lamberteschi) -> 0.00000000
```

```
(Ginori, Lamberteschi) -> 0.00000000
```

In [8]:

```
# -- plot Florentine Families graph
nx.draw_networkx_nodes(G, nodelist=nodes, label=nodes, pos=layout, node_s
nx.draw_networkx_edges(G, edgelist=old_edges, pos=layout, edge_color='gra


# add all edges containing ginori to a list
# add all similarity of ginori with neighbours to a list
new_edges_ginori = []
metric_ginori = []
for i, term in enumerate(new_edges):
    if 'Ginori' in term:
        new_edges_ginori.append(new_edges[i])
        metric_ginori.append(metric[i])

# -- plot edges representing similarity
"""
    This example is randomly plotting similarities between 8 pairs of noc
    Identify the "Ginori"
"""
ne = nx.draw_networkx_edges(G, edgelist=new_edges_ginori, pos=layout, edg
plt.colorbar(ne)
plt.axis('off')
plt.show()
```